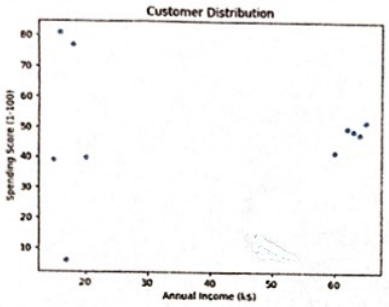
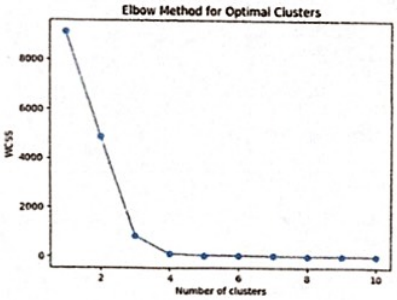
9

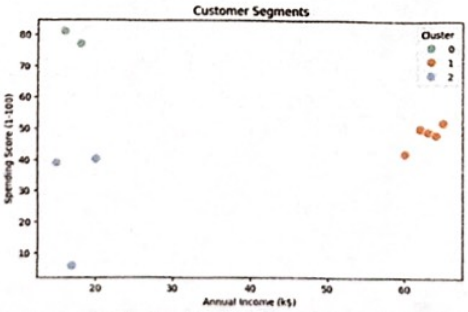
```
import pandas as pd
import matplotlib.pyplot as plt
import Seaborn as sns
from sklearn.cluster import KMeans

# Load or simulate customer data
# Sample dataset : Annual Income vs Spending Score
data = {
    'Customer ID' : [1,2,3,4,5,6,7,8,9,10],
    'Annual Income (K$)' : [15,16,17,18,20,60,62,63,64,65],
    'Spending Score (1-100)' : [39,81,6,77,40,42,50,49,48,52]
}
df = pd.DataFrame(data)
# select features for clustering
X = df[['Annual Income (K$)', 'Spending Score (1-100)']]
# visualize data before clustering
sns.scatterplot(X='Annual Income (K$)', y='Spending Score (1-100)',
    data = df)
plt.title('Customer Distribution')
plt.show()
Use the Elbow Method to find optimal number of clusters
wcss = []
for i in range(1,11):
    ~~Kmo~~ Kmeans = KMeans(n_clusters = i, init = 'k-means++',
                            random_state = 42)
    Kmeans.fit(X)
```

```
wcss.append(kmeans.inertia_)
plt.plot(range(1,11), wcss, marker='o')
plt.title('Elbow Method for optimal Clusters')
plt.xlabel('Number of clusters')
plt.ylabel('wcss')
plt.show()

#Apply K-Means with optimal clusters (eg, 3)
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=42)
df['cluster'] = kmeans.fit_predict(X)
#visualize clustered groups
plt.figure(figsize=(8,5))
sns.scatterplot(
  x='Annual Income (K$)', y='Spending Score (1-100)',
  hue='cluster', palette='Set2', data=df, s=100
)
plt.title('customer Segments')
plt.show()
```

Output for 9


Customer Distribution
Spending Score (1-100)
Annual Income (k$)


Elbow Method for Optimal Clusters
WCSS
Number of clusters

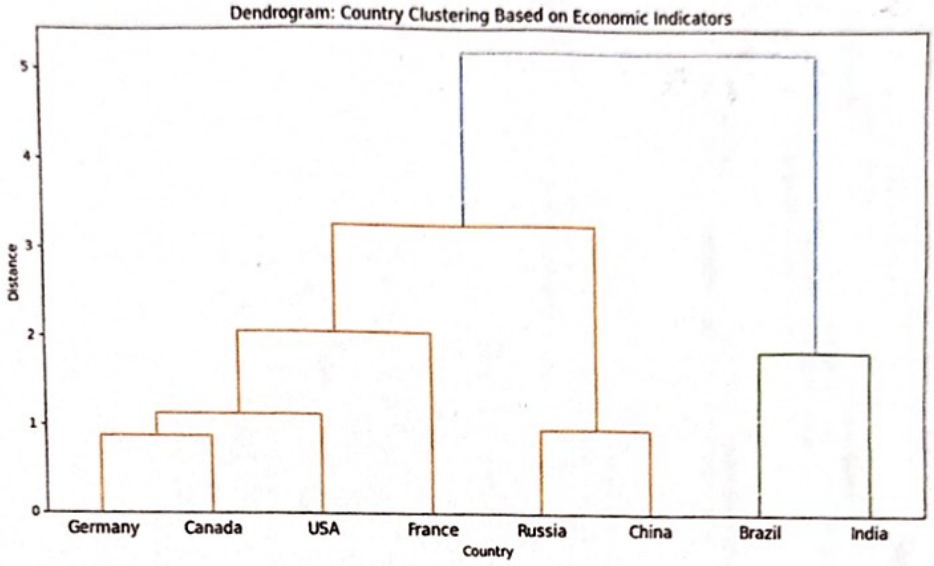

Customer Segments
Cluster
0
1
2
Spending Score (1-100)
Annual Income (k$)

10

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster
from sklearn.preprocessing import StandardScaler
#Sample data : Replace with real-world data as needed
data = {
    'Country': ['USA', 'Canada', 'Germany', 'France', 'India', 'China', 'Brazil',
                'Russia'],
    'GDP Per Capita': [63000, 46000, 48000, 41000, 2100, 12000, 8800, 11400],
    'Inflation Rate': [2.3, 1.5, 1.4, 1.8, 5.5, 2.1, 4.2, 3.4],
    'Unemployment Rate': [5.2, 6.0, 4.5, 8.0, 7.1, 5.0, 9.8, 4.8]}
df = pd.DataFrame(data)
X = df[['GDP per Capita', 'Inflation Rate', 'Unemployment Rate']]
#Standardize features (important for distance metrics)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
#Perform hierarchical clustering using Ward's method
linked = linkage(X_scaled, method='ward')
#Plot dendrogram
plt.figure(figsize=(10,6))
dendrogram(linked, labels = df['Country'].values, orientation='top',
distance_sort='descending', show_leaf_counts=True)
```

```
plt.title('Dendrogram : Country clustering Based on Economic indicators')
plt.xlabel('Country')
plt.ylabel('Euclidean Distance')
plt.tight_layout()
plt.show()

# optional : assign clusters
df['cluster'] = fcluster(linked, t=3, criterion='maxclust')

# Display clustered data
print(df[['Country', 'Cluster']])
```

Output for 10



	Country	Cluster
0	USA	3
1	Canada	3
2	Germany	3
3	France	3
4	India	1
5	China	2
6	Brazil	1
7	Russia	2